Payment Rules through Discriminant-Based Classifiers

Payment Rules through Discriminant-Based Classifiers

PAUL DÜTTING, École Polytechnique Fédérale de Lausanne, Switzerland
FELIX FISCHER, University of Cambridge, UK
PICHAYUT JIRAPINYO, Harvard University, USA
JOHN K. LAI, Harvard University, USA
BENJAMIN LUBIN, Boston University, USA
DAVID C. PARKES, Harvard University, USA

In mechanism design it is typical to impose incentive compatibility and then derive an optimal mechanism subject to this constraint. By replacing the incentive compatibility requirement with the goal of minimizing expected ex post regret, we are able to adapt statistical machine learning techniques to the design of payment rules. This computational approach to mechanism design is applicable to domains with multi-dimensional types and situations where computational efficiency is a concern. Specifically, given an outcome rule and access to a type distribution, we train a support vector machine with a special discriminant function structure such that it implicitly establishes a payment rule with desirable incentive properties. We discuss applications to a multi-minded combinatorial auction with a greedy winner-determination algorithm and to an assignment problem with egalitarian outcome rule. Experimental results demonstrate both that the construction produces payment rules with low ex post regret, and that penalizing classification errors is effective in preventing failures of ex post individual rationality.

Categories and Subject Descriptors: J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Economics*; I.2.6 [**Artificial Intelligence**]: Learning

General Terms: Algorithms, Economics, Theory

Additional Key Words and Phrases: Computational Mechanism Design, Support Vector Machines

1. INTRODUCTION

Mechanism design studies situations where a set of agents each hold private information about their preferences over different outcomes. The designer chooses a center that receives claims about such preferences, selects and enforces an outcome, and optionally collects payments. The classical approach is to impose *incentive compatibility*, ensuring that agents truthfully report their preferences in strategic equilibrium. Subject to this constraint, the goal is to identify a mechanism, i.e., a way of choosing an outcome and payments based on agents' reports, that optimizes a given design objective like social welfare, revenue, or some notion of fairness.

There are, however, significant challenges associated with this classical approach. First of all, it can be analytically cumbersome to derive optimal mechanisms for domains that are "multi-dimensional" in the sense that each agent's private information is described through more than a single number, and few results are known in this case.[1] Second, incentive compatibility can be costly, in that adopting it as a hard con-

[1]One example of a multi-dimensional domain is a combinatorial auction, where an agent's preferences are described by a numerical value for each of several different bundles of items.

EC'12, June 4–8, 2012, Valencia, Spain.

straint can preclude mechanisms with useful economic properties. For example, imposing the strongest form of incentive compatibility, truthfulness in a dominant strategy equilibrium or *strategyproofness*, necessarily leads to poor revenue, vulnerability to collusion, and vulnerability to false-name bidding in combinatorial auctions where valuations exhibit complementarities among items [Ausubel and Milgrom 2006; Rastegari et al. 2011]. A third difficulty occurs when the optimal mechanism has an outcome or payment rule that is computationally intractable.

In the face of these difficulties, we adopt statistical machine learning to automatically infer mechanisms with good incentive properties. Rather than imposing incentive compatibility as a hard constraint, we start from a given outcome rule and use machine learning techniques to identify a payment rule that minimizes agents' *expected ex post regret* relative to this outcome rule. Here, the ex post regret an agent has for truthful reporting in a given instance is the amount by which its utility could be increased through a misreport. While a mechanism with zero ex post regret for all inputs is obviously strategyproof, we are not aware of any additional direct implication in terms of equilibrium properties.[2] Support for expected ex post regret as a quantifiable target for mechanism design rather comes from a simple model of manipulation where agents face a certain cost for strategic behavior. If this cost is higher than the expected gain, agents can be assumed to behave truthfully. We do insist on mechanisms in which the price to an agent, conditioned on an outcome, is independent of its report. This provides additional robustness against manipulation in the sense that there is no local price sensitivity.[3]

Our approach is applicable to domains that are multi-dimensional or for which the computational efficiency of outcome rules is a concern. Given the implied relaxation of incentive compatibility, the intended application is to domains in which incentive compatibility is unavailable or undesirable for outcome rules that meet certain economic and computational desiderata. The payment rule is learned on the basis of a given outcome rule, and as such the framework is most meaningful in domains where revenue considerations are secondary to outcome considerations.

The essential insight is that the payment rule of a strategyproof mechanism can be thought of as a classifier for predicting the outcome: the payment rule implies a price to each agent for each outcome, and the selected outcome must be one that simultaneously maximizes reported value minus price for every agent. By limiting classifiers to discriminant functions[4] with this "value-minus-price" structure, where the price can be an arbitrary function of the outcome and the reports of other agents, we obtain a remarkably direct connection between multi-class classification and mechanism design.

---

[2]The expected ex post regret given a distribution over types provides an upper bound on the expected regret of an agent who knows its own type but has only distributional information on the types of other agents. The latter metric is also appealing, but does not seem to fit well with the generalization error of statistical machine learning. An emerging literature is developing various regret-based metrics for quantifying the incentive properties of mechanisms [Parkes et al. 2001; Day and Milgrom 2008; Lubin 2010; Carroll 2011], and there also exists experimental support for a quantifiable measure of the *divergence* between the distribution on payoffs in a mechanism and that in a strategyproof reference mechanism like the VCG mechanism [Lubin and Parkes 2009]. An earlier literature had looked for approximate incentive compatibility or incentive compatibility in the large-market limit, see, e.g., the recent survey by Carroll [2011]. Related to the general theme of relaxing incentive compatibility is work of Pathak and Sönmez [2010] that provides a qualitative ranking of different mechanisms in terms of the number of manipulable instances, and work of Budish [2010] that introduces an asymptotic, binary, design criterion regarding incentive properties in a large replica economy limit. Whereas the present work is constructive, the latter seek to explain which mechanisms are adopted in practice.

[3]Erdil and Klemperer [2010] consider a metric that emphasizes this property.

[4]A discriminant function can be thought of as a way to distinguish between different outcomes for the purpose of making a prediction.

For an appropriate loss function, the discriminant function of a classifier that minimizes generalization error over a hypothesis class has a corresponding payment rule that minimizes expected ex post regret among all payment rules corresponding to classifiers in this class. Conveniently, an appropriate method exists for multi-class classification with large outcome spaces that supports the specific structure of the discriminant function, namely the method of *structural support vector machines* [Tsochantaridis et al. 2005; Joachims et al. 2009]. Just like standard support vector machines, it allows us to adopt non-linear kernels, thus enabling price functions that depend in a non-linear way on the outcome and on the reported types of other agents.

In illustrating the framework, we focus on two situations where strategyproof payment rules are not available: a greedy outcome rule for a multi-minded combinatorial auction in which each agent is interested in a constant number of bundles, and an assignment problem with an egalitarian outcome rule, i.e., an outcome rule that maximizes the minimum value of any agent. The experimental results we obtain are encouraging, in that they demonstrate low expected ex post regret even when the $0/1$ classification accuracy is only moderately good, and in particular better regret properties than those obtained through simple VCG-based payment rules that we adopt as a baseline. In addition, we give special consideration to the failure of ex post individual rationality, and introduce methods to bias the classifier to avoid these kinds of errors as well as post hoc adjustments that eliminate them. As far as scalability is concerned, we emphasize that the computational cost associated with our approach occurs offline during training. The learned payment rules have a succinct description and can be evaluated quickly in a deployed mechanism.

Due to space constraints, we omit some extend examples, proofs, and tables, and refer the reader to the full version of the paper for details.

Related Work

Conitzer and Sandholm [2002] introduced the agenda of *automated mechanism design* (AMD), which formulates mechanism design as an optimization problem. The output is the *description* of a mechanism, i.e., an explicit mapping from types to outcomes and payments. AMD is intractable in general, as the type space can be exponential in both the number of agents and the number of items, but progress has recently been made in finding approximate solutions for domains with additive value structure and symmetry assumptions, and adopting Bayes-Nash incentive compatibility (BIC) as the goal [Cai et al. 2012]. Another approach is to search through a parameterized space of incentive-compatible mechanisms [Guo and Conitzer 2010].

A parallel literature allows outcome rules to be represented by *algorithms*, like our work, and thus extends to richer domains. Lavi and Swamy [2005] employ LP relaxation to obtain mechanisms satisfying BIC for set-packing problems, achieving worst-case approximation guarantees for combinatorial auctions. Hartline and Lucier [2010] and Hartline et al. [2011] propose a general approach, applicable to both single-parameter and multi-parameter domains, for converting any approximation algorithm into a mechanism satisfying BIC that has essentially the same approximation factor with respect to social welfare. This approach differs from ours in that it adopts BIC as a target rather than the minimization of expected ex post regret. In addition, it evaluates the outcome rule on a number of randomly perturbed replicas of the instance that is polynomial in the size of a discrete type space, which is infeasible for combinatorial auctions where this size is exponential in the number of items. The computational requirements of our trained rule are equivalent to that of the original outcome rule.

Lahaie [2009, 2010] also adopts a kernel-based approach for combinatorial auctions, but focuses not on learning a payment rule for a given outcome rule but rather on solving the winner determination and pricing problem for a given instance of a combi-

natorial auction. Lahaie introduces the use of kernel methods to compactly represent non-linear price functions, which is also present in our work, but obtains incentive properties more indirectly through a connection between regularization and price sensitivity.

2. PRELIMINARIES

A mechanism design problem is given by a set $N = \{1, 2, \ldots, n\}$ of *agents* that interact to select an element from a set $\Omega \subseteq \times_{i \in N} \Omega_i$ of *outcomes*, where Ω_i denotes the set of possible outcomes for agent $i \in N$. Agent $i \in N$ is associated with a *type* θ_i from a set Θ_i of possible types, corresponding to the private information available to this agent. We write $\theta = (\theta_1, \ldots, \theta_n)$ for a profile of types for the different agents, $\Theta = \times_{i \in N} \Theta_i$ for the set of possible type profiles, and $\theta_{-i} \in \Theta_{-i}$ for a profile of types for all agents but i. Each agent $i \in N$ is further assumed to employ preferences over Ω_i, represented by a *valuation function* $v_i : \Theta_i \times \Omega_i \to \mathbb{R}$. We assume that for all $i \in N$ and $\theta_i \in \Theta_i$ there exists an outcome $o \in \Omega$ with $v_i(\theta_i, o_i) = 0$.

A *(direct) mechanism* is a pair (g, p) of an *outcome rule* $g : \Theta \to \times_{i \in N} \Omega_i$ and a *payment rule* $p : \Theta \to \mathbb{R}^n_{\geq 0}$. The intuition is that the agents reveal to the mechanism a type profile $\theta \in \Theta$, possibly different from their true types, and the mechanism chooses outcome $g(\theta)$ and charges each agent i a payment of $p_i(\theta) = (p(\theta))_i$. We assume *quasi-linear preferences*, so the *utility* of agent i with type $\theta_i \in \Theta_i$ given a profile $\theta' \in \Theta$ of revealed types is $u_i(\theta', \theta_i) = v_i(\theta_i, g_i(\theta')) - p_i(\theta')$, where $g_i(\theta) = (g(\theta)_i)$ denotes the outcome for agent i. A crucial property of mechanism (g, p) is that its outcome rule is *feasible*, i.e., that $g(\theta) \in \Omega$ for all $\theta \in \Theta$.

Outcome rule g satisfies *consumer sovereignty* if for all $i \in N$, $o_i \in \Omega_i$, and $\theta'_{-i} \in \Theta_{-i}$, there exists $\theta'_i \in \Theta_i$ such that $g_i(\theta'_i, \theta'_{-i}) = o_i$; and *reachability of the null outcome* if for all $i \in N$, $\theta_i \in \Theta_i$, and $\theta'_{-i} \in \Theta_{-i}$, there exists $\theta'_i \in \Theta_i$ such that $v_i(\theta_i, g_i(\theta'_i, \theta'_{-i})) = 0$.

Mechanism (g, p) is *dominant strategy incentive compatible*, or *strategyproof*, if each agent maximizes its utility by reporting its true type, irrespective of the reports of the other agents, i.e., if for all $i \in N$, $\theta_i \in \Theta_i$, and $\theta' = (\theta'_i, \theta'_{-i}) \in \Theta$, $u_i((\theta_i, \theta'_{-i}), \theta_i) \geq u_i((\theta'_i, \theta'_{-i}), \theta_i)$; it satisfies *individual rationality* (IR) if agents reporting their true types are guaranteed non-negative utility, i.e., if for all $i \in N$, $\theta_i \in \Theta_i$, and $\theta'_{-i} \in \Theta_{-i}$, $u_i((\theta_i, \theta'_{-i}), \theta_i) \geq 0$. Observe that given reachability of the null outcome, strategyproofness implies individual rationality.

It is known that a mechanism (g, p) is strategyproof if and only if the payment of an agent is independent of its reported type and the chosen outcome simultaneously maximizes the utility of all agents, i.e., if for every $\theta \in \Theta$,

$$p_i(\theta) = t_i(\theta_{-i}, g_i(\theta)) \qquad \text{for all } i \in N, \text{ and} \tag{1}$$

$$g_i(\theta) \in \arg\max_{o'_i \in \Omega_i} \big(v_i(\theta_i, o'_i) - t_i(\theta_{-i}, o'_i)\big) \qquad \text{for all } i \in N, \tag{2}$$

for a *price function* $t_i : \Theta_{-i} \times \Omega_i \to \mathbb{R}$. This simple characterization is crucial for the main results in the present paper, providing the basis with which the discriminant function of a classifier can be used to induce a payment rule.

In addition, a direct characterization of strategyproofness in terms of monotonicity properties of outcome rules explains which outcome rules can be associated with a payment rule in order to be "implementable" within a strategyproof mechanism [Saks and Yu 2005; Ashlagi et al. 2010]. These monotonicity properties provide a fundamental constraint on when our machine learning framework can hope to identify a payment rule that provides full strategyproofness.

We quantify the degree of strategyproofness of a mechanism in terms of the *regret* experienced by an agent when revealing its true type, i.e., the potential gain in utility by revealing a different type instead. Formally, the *ex post regret* of agent $i \in N$ in

mechanism (g, p), given true type $\theta_i \in \Theta_i$ and reported types $\theta'_{-i} \in \Theta_{-i}$ of the other agents, is

$$rgt_i(\theta_i, \theta'_{-i}) = \max_{\theta'_i \in \Theta_i} u_i\big((\theta'_i, \theta'_{-i}), \theta_i\big) - u_i\big((\theta_i, \theta'_{-i}), \theta_i\big).$$

Analogously, the *ex post violation of individual rationality* of agent $i \in N$ in mechanism (g, p), given true type $\theta_i \in \Theta_i$ and reported types $\theta'_{-i} \in \Theta_{-i}$ of the other agents, is

$$irv_i(\theta_i, \theta'_{-i}) = |\min(u_i((\theta_i, \theta'_{-i}), \theta_i), 0)|.$$

We consider situations where types are drawn from a distribution with probability density function $D : \Theta \to \mathbb{R}$ such that $D(\theta) \geq 0$ and $\int_{\theta \in \Theta} D(\theta) = 1$. Given such a distribution, and assuming that all agents report their true types, the *expected ex post regret* of agent $i \in N$ in mechanism (g, p) is $\mathbb{E}_{\theta \sim D}[rgt_i(\theta_i, \theta_{-i})]$.

Outcome rule g is *agent symmetric* if for every permutation π of N and all types $\theta, \theta' \in \Theta$ such that $\theta_i = \theta'_{\pi(i)}$ for all $i \in N$, $g_i(\theta) = g_{\pi(i)}(\theta')$ for all $i \in N$. Note that this specifically requires that $\Theta_i = \Theta_j$ and $\Omega_i = \Omega_j$ for all $i, j \in N$. Similarly, type distribution D is *agent symmetric* if $D(\theta) = D(\theta')$ for every permutation π of N and all types $\theta, \theta' \in \Theta$ such that $\theta_i = \theta'_{\pi(i)}$ for all $i \in N$. Given agent symmetry, a price function $t_1 : \Theta_{-1} \times \Omega_i \to \mathbb{R}$ for agent 1 can be used to generate the payment rule p for a mechanism (g, p), with

$$p(\theta) = \big(t_1(\theta_{-1}, g_1(\theta)), t_1(\theta_{-2}, g_2(\theta)), \ldots, t_1(\theta_{-n}, g_n(\theta))\big),$$

so that the expected ex post regret is the same for every agent.

We assume agent symmetry in the sequel, which precludes outcome rules that break ties based on agent identity, but obviates the need to train a separate classifier for each agent while also providing some benefits in terms of presentation. Because ties occur only with negligible probability in our experimental framework, the experimental results are not affected by this assumption.

3. PAYMENT RULES FROM MULTI-CLASS CLASSIFIERS

A *multi-class classifier* is a function $h : X \to Y$, where X is an input domain and Y is a discrete output domain. One could imagine, for example, a multi-class classifier that labels a given image as that of a dog, a cat, or some other animal. In the context of mechanism design, we will be interested in classifiers that take as input a type profile and output an outcome. What distinguishes this from an outcome rule is that we will impose restrictions on the form the classifier can take.

Classification typically assumes an underlying target function $h^* : X \to Y$, and the goal is to learn a classifier h that minimizes disagreements with h^* on a given input distribution D on X, based only on a finite set of *training data* $\{(x^1, y^1), \ldots, (x^\ell, y^\ell)\} = \{(x^1, h^*(x^1)), \ldots, (x^\ell, h^*(x^\ell))\}$ with $x^1, \ldots, x^\ell$ drawn from D. This may be challenging because the amount of training data is limited, or because h is restricted to some hypothesis class $\mathcal{H}$ with a certain simple structure, e.g., linear threshold functions. If $h(x) = h^*(x)$ for all $x \in X$, we say that h is a *perfect classifier* for h^*.

We consider classifiers that are defined in terms of a *discriminant function* $f : X \times Y \to \mathbb{R}$, such that

$$h(x) \in \arg\max_{y \in Y} f(x, y)$$

for all $x \in X$. More specifically, we will be concerned with *linear* discriminant functions of the form

$$f_w(x, y) = w^T \psi(x, y)$$

for a weight vector $w \in \mathbb{R}^m$ and a *feature map* $\psi : X \times Y \to \mathbb{R}^m$, where $m \in \mathbb{N} \cup \{\infty\}$.[5] The function ψ maps input and output into an m-dimensional space, which generally allows non-linear features to be expressed.

3.1. Mechanism Design as Classification

Assume that we are given an outcome rule g and access to a distribution D over type profiles, and want to design a corresponding payment rule p that gives the mechanism (g, p) the best possible incentive properties. Assuming agent symmetry, we focus on a partial outcome rule $g_1 : \Theta \to \Omega_1$ and train a classifier to predict the outcome to agent 1. To train a classifier, we generate examples by drawing a type profile $\theta \in \Theta$ from distribution D and applying outcome rule g to obtain the target class $g_1(\theta) \in \Omega_1$.

We impose a special structure on the hypothesis class. A classifier $h_w : \Theta \to \Omega_1$ is *admissible* if it is defined in terms of a discriminant function f_w of the form

$$f_w(\theta, o_1) = w_1 v_1(\theta_1, o_1) + w_{-1}^T \psi(\theta_{-1}, o_1)$$

for weights w such that $w_1 \in \mathbb{R}_{>0}$ and $w_{-1} \in \mathbb{R}^m$, and a feature map $\psi : \Theta_{-1} \times \Omega_1 \to \mathbb{R}^m$ for $m \in \mathbb{N} \cup \{\infty\}$.

The first term of $f_w(\theta, o_1)$ only depends on the type of agent 1 and increases in its valuation for outcome o_1, while the remaining terms ignore θ_1 entirely. This restriction allows us to directly infer agent-independent prices from a trained classifier. For this, define the *associated price function* of an admissible classifier h_w as

$$t_w(\theta_{-1}, o_1) = -\frac{1}{w_1} w_{-1}^T \psi(\theta_{-1}, o_1),$$

where we again focus on agent 1 for concreteness. By agent symmetry, we obtain the mechanism (g, p_w) corresponding to classifier h_w by letting

$$p_w(\theta) = \big(t_w(\theta_{-1}, g_1(\theta)), t_w(\theta_{-2}, g_2(\theta)), \ldots, t_w(\theta_{-n}, g_n(\theta))\big).$$

Even with admissibility, appropriate choices for the feature map ψ will produce rich families of classifiers, and thus ultimately useful payment rules. Moreover, this form is compatible with structural support vector machines, discussed in Section 4.1.

3.2. Example: Single-Item Auction

Before proceeding further, we illustrate the ideas developed so far in the context of a single-item auction. In a single-item auction, the type of each agent is a single number, corresponding to its value for the item being auctioned, and there are two possible allocations from the point of view of agent 1: one where it receives the item, and one where it does not. Formally, $\Theta = \mathbb{R}^n$ and $\Omega_1 = \{0, 1\}$.

Consider a setting with three agents and a training set

$$(\theta^1, o_1^1) = ((1, 3, 5), 0), \quad (\theta^2, o_1^2) = ((5, 4, 3), 1), \quad (\theta^3, o_1^3) = ((2, 3, 4), 0),$$

and note that this training set is consistent with an *optimal* outcome rule, i.e., one that assigns the item to an agent with maximum value. Our goal is to learn an admissible classifier

$$h_w(\theta) = \arg\max_{o_1 \in \{0,1\}} f_w(\theta, o_1) = \arg\max_{o_1 \in \{0,1\}} w_1 v_1(\theta_1, o_1) + w_{-1}^T \psi(\theta_{-1}, o_1)$$

that performs well on the training set. Since there are only two possible outcomes, the outcome chosen by h_w is simply the one with the larger discriminant. A classifier that is perfect on the training data must therefore satisfy the following constraints:

$$w_1 \cdot 0 + w_{-1}^T \psi((3, 5), 0) > w_1 \cdot 1 + w_{-1}^T \psi((3, 5), 1),$$

[5] We allow w to have infinite dimension, but require the inner product between w and $\psi(x, y)$ to be defined in any case. Computationally the infinite-dimensional case is handled through the kernel trick, which is described in Section 4.1.1.

$$w_1 \cdot 5 + w_{-1}^T \psi((4,3),1) > w_1 \cdot 0 + w_{-1}^T \psi((4,3),0),$$
$$w_1 \cdot 0 + w_{-1}^T \psi((3,4),0) > w_1 \cdot 2 + w_{-1}^T \psi((3,4),1).$$

This can for example be achieved by setting $w_1 = 1$ and

$$w_{-1}^T \psi((\theta_2,\theta_3),o_1) = \begin{cases} -\max(\theta_2,\theta_3) & \text{if } o_1 = 1 \text{ and} \\ 0 & \text{if } o_1 = 0. \end{cases} \tag{3}$$

Recalling our definition of the price function as $t_w(\theta_{-1},o_1) = -(1/w_1)w_{-1}^T\psi(\theta_{-1},o_1)$, we see that this choice of w and ψ corresponds to the second-price payment rule. We will see in the next section that this relationship is not a coincidence.[6]

3.3. Perfect Classifiers and Implementable Outcome Rules

We now formally establish a connection between implementable outcome rules and perfect classifiers.

THEOREM 3.1. *Let (g,p) be a strategyproof mechanism with an agent symmetric outcome rule g, and let t_1 be the corresponding price function. Then, a perfect admissible classifier h_w for partial outcome rule g_1 exists if $\arg\max_{o_1\in\Omega_1}(v_1(\theta_1,o_1) - t_1(\theta_{-1},o_1)))$ is unique.*

PROOF. By the first characterization of strategyproof mechanisms, g must select an outcome that maximizes the utility of agent 1 at the current prices, i.e.,

$$g_1(\theta) \in \arg\max_{o_1\in\Omega_1}(v_1(\theta_i,o_1) - t_1(\theta_{-1},o_1)).$$

Consider the admissible discriminant $f_{(1,1)}(\theta,o_1) = v_1(\theta_1,o_1) - t_1(\theta_{-1},o_1)$, which uses the price function t_1 as its feature map. Clearly, the corresponding classifier $h_{(1,1)}$ maximizes the same quantity as g_1, and the two must agree if there is a unique maximizer. □

The relationship also works in the opposite direction: a perfect, admissible classifier h_w for outcome rule g can be used to construct a payment rule that turns g into a strategyproof mechanism.

THEOREM 3.2. *Let g be an agent symmetric outcome rule, $h_w : \Theta \to \Omega_1$ an admissible classifier, and p_w the payment rule corresponding to h_w. If h_w is a perfect classifier for the partial outcome rule g_1, then the mechanism (g,p_w) is strategyproof.*

We prove this result by expressing the regret of an agent in mechanism (g,p_w) in terms of the discriminant function f_w. Let $\Omega_i(\theta_{-i}) \subseteq \Omega_i$ denote the set of partial outcomes for agent i that can be obtained under g given reported types θ_{-i} from all agents but i, keeping the dependence on g silent for notational simplicity.

LEMMA 3.3. *Suppose that agent 1 has type θ_1 and that the other agents report types θ_{-1}. Then the regret of agent 1 for bidding truthfully in mechanism (g,p_w) is*

$$\frac{1}{w_1}\big(\max_{o_1\in\Omega(\theta_{-1})} f_w(\theta,o_1) - f_w(\theta,g_1(\theta))\big).$$

[6] In practice, we are limited in the machine learning framework to hypotheses that are linear in $\psi((\theta_2,\theta_3),o_1)$, and will not be able to guarantee that (3) holds exactly. In Section 4.1.1 we will see, however, that certain choices of ψ allow for very complex hypotheses that can closely approximate arbitrary functions.

PROOF. We have

$$\begin{aligned} rgt_1(\theta) &= \max_{\theta_1' \in \Theta_1} \big(v_1(\theta_1, g_1(\theta_1', \theta_{-1})) - p_{w,1}(\theta_1', \theta_{-1})\big) - \big(v_1(\theta_1, g_1(\theta)) - p_{w,1}(\theta)\big) \\ &= \max_{o_1 \in \Omega_1(\theta_{-1})} \big(v_1(\theta_1, o_1) - t_w(\theta_{-1}, o_1)\big) - \big(v_1(\theta_1, g_1(\theta)) - t_w(\theta_{-1}, g_1(\theta))\big) \\ &= \max_{o_1 \in \Omega_1(\theta_{-1})} \big(v_1(\theta_1, o_1) + \frac{1}{w_1} w_{-1}^T \psi(\theta_{-1}, o_1)\big) - \\ &\qquad\qquad \big(v_1(\theta_1, g_1(\theta)) + \frac{1}{w_1} w_{-1}^T \psi(\theta_{-1}, g_1(\theta))\big) \\ &= \frac{1}{w_1} \big(\max_{o_1 \in \Omega_1(\theta_{-1})} f_w(\theta, o_1) - f_w(\theta, g_1(\theta))\big). \end{aligned}$$

□

PROOF OF THEOREM 3.2. If h_w is a perfect classifier, then the discriminant function f_w satisfies $\arg\max_{o_1 \in \Omega_1} f_w(\theta, o_1) = g_1(\theta)$ for every $\theta \in \Theta$. Since $g_1(\theta) \in \Omega_1(\theta_{-1})$, we thus have that $\max_{o_1 \in \Omega_1(\theta_{-1})} f_w(\theta, o_1) = f_w(\theta, g_1(\theta))$. By Lemma 3.3, the regret of agent 1 for bidding truthfully in mechanism (g, p_w) is always zero, which means that the mechanism is strategyproof. □

It bears emphasis that classifier h_w is only used to derive the payment rule p_w, while the outcome is still selected according to g. In principle, classifier h_w could be used to obtain an agent symmetric outcome rule g_w and, since h_w is a perfect classifier for itself, a strategyproof mechanism (g_w, p_w). Unfortunately, outcome rule g_w is not in general feasible. Mechanism (g, p_w), on the other hand, is not strategyproof when h_w fails to be a perfect classifier for g. While payment rule p_w always satisfies the agent-independence property (1) required for strategyproofness, the "optimization" property (2) might be violated when $h_w(\theta) \neq g_1(\theta)$.

3.4. Approximate Classification and Approximate Strategyproofness

A perfect admissible classifier for outcome rule g leads to a payment rule that turns g into a strategyproof mechanism. We now show that this result extends gracefully to situations where no such payment rule is available, by relating the *expected* ex post regret of a mechanism (g, p) to a measure of the generalization error of a classifier for g.

Fix a feature map ψ, and denote by $\mathcal{H}_\psi$ the space of all admissible classifiers with this feature map. The *discriminant loss* of a classifier $h_w \in \mathcal{H}_\psi$ with respect to a type profile θ and an outcome $o_1 \in \Omega_1$ is given by

$$\Delta_w(o_1, \theta) = \frac{1}{w_1} \big(f_w(\theta, h_w(\theta)) - f_w(\theta, o_1)\big).$$

Intuitively the discriminant loss measures how far, in terms of the normalized discriminant, h_w is from predicting the correct outcome for type profile θ, assuming the correct outcome is o_1. Note that $\Delta(o_1, \theta) \geq 0$ for all $o_1 \in \Omega_1$ and $\theta \in \Theta$, and $\Delta(o_1, \theta) = 0$ if $o_1 = h_w(\theta)$. Note further that $h_w(\theta) = h_{w'}(\theta)$ does not imply that $\Delta_w(o_1, \theta) = \Delta_{w'}(o_1, \theta)$ for all $o_1 \in \Omega_1$: even if two classifiers predict the same outcome, one of them may still be closer to predicting the correct outcome o_1.

The *generalization error* of classifier $h_w \in \mathcal{H}_\psi$ with respect to a type distribution D and a partial outcome rule $g_1 : \Theta \to \Omega_1$ is then given by

$$R_w(D, g) = \int_{\theta \in \Theta} \Delta_w\big(g_1(\theta), \theta\big) D(\theta) d\theta.$$

The following result establishes a connection between the generalization error and the expected ex post regret of the corresponding mechanism.

THEOREM 3.4. *Consider an outcome rule g, a space $\mathcal{H}_\psi$ of admissible classifiers, and a type distribution D. Let $h_{w^*} \in \mathcal{H}_\psi$ be a classifier that minimizes generalization error with respect to D and g among all classifiers in $\mathcal{H}_\psi$. Then the following holds:*

(*1*) *If g satisfies consumer sovereignty, then (g, p_{w^*}) minimizes expected ex post regret with respect to D among all mechanisms (g, p_w) corresponding to classifiers $h_w \in \mathcal{H}_\psi$.*
(*2*) *Otherwise, (g, p_{w^*}) minimizes an upper bound on expected ex post regret with respect to D amongst all mechanisms (g, p_w) corresponding to classifiers $h_w \in \mathcal{H}_\psi$.*

PROOF. For the second property, observe that

$$\begin{aligned}\Delta_w(g_1(\theta), \theta) &= \frac{1}{w_1}\big(f_w(\theta, h_w(\theta)) - f_w(\theta, g_1(\theta))\big) \\ &= \frac{1}{w_1}\big(\max_{o_1 \in \Omega_1} f_w(\theta, o_1) - f_w(\theta, g_1(\theta))\big) \\ &\geq \frac{1}{w_1}\big(\max_{o_1 \in \Omega(\theta_{-1})} f_w(\theta, o_1) - f_w(\theta, g_1(\theta))\big) = rgt_1(\theta),\end{aligned}$$

where the last equality holds by Lemma 3.3. If g satisfies consumer sovereignty, then the inequality holds with equality, and the first property follows as well. □

Minimization of expected regret itself, rather than an upper bound, can also be achieved if the learner has access to the set $\Omega_1(\theta_{-1})$ for every $\theta_{-1} \in \Theta_{-1}$.

4. A SOLUTION USING STRUCTURAL SUPPORT VECTOR MACHINES

In this section we discuss the method of *structural support vector machines* (structural SVMs) [Tsochantaridis et al. 2005; Joachims et al. 2009], and show how it can be adapted for the purpose of learning classifiers with admissible discriminant functions.

4.1. Structural SVMs

Given an input space X, a discrete output space Y, a target function $h^* : X \to Y$, and a set of *training examples* $\{(x^1, h^*(x^1)), \ldots, (x^\ell, h^*(x^\ell))\} = \{(x^1, y^1), \ldots, (x^\ell, y^\ell)\}$, structural SVMs learn a multi-class classifier h that on input $x \in X$ selects an output $y \in Y$ that maximizes $f_w(x, y) = w^T \psi(x, y)$. For a given feature map ψ, the training problem is to find a vector w for which h_w has low generalization error.

Given examples $\{(x^1, y^1), \ldots, (x^\ell, y^\ell)\}$, training is achieved by solving the following convex optimization problem:

$$\min_{w, \xi \geq 0} \ \frac{1}{2} w^T w + \frac{C}{\ell} \sum_{k=1}^{\ell} \xi^k \qquad \text{(Training Problem 1)}$$

$$\text{s.t.} \ \ w^T\big(\psi(x^k, y^k) - \psi(x^k, y)\big) \geq \mathcal{L}(y^k, y) - \xi^k \quad \text{for all } k = 1, \ldots, \ell,\ y \in Y$$

$$\xi^k \geq 0 \quad \text{for all } k = 1, \ldots, \ell.$$

The goal is to find a weight vector w and slack variables ξ^k such that the objective function is minimized while satisfying the constraints. The learned weight vector w parameterizes the discriminant function f_w, which in turn defines the classifier h_w. The kth constraint states that the value of the discriminant function on (x^k, y^k) should exceed the value of the discriminant function on (x^k, y) by at least $\mathcal{L}(y^k, y)$, where $\mathcal{L}$ is a loss function that penalizes misclassification, with $\mathcal{L}(y, y) = 0$ and $\mathcal{L}(y, y') \geq 0$ for all $y, y' \in Y$. We generally use a $0/1$ loss function, but consider an alternative in Section 4.2.2 to improve ex post IR properties. Positive values for the slack variables ξ^k allow the weight vector to violate some of the constraints.

The other term in the objective, the squared norm of w, penalizes scaling of w. This is necessary because scaling of w can arbitrarily increase the margin between $f_w(x^k, y^k)$ and $f_w(x^k, y)$ and make the constraints easier to satisfy. Smaller values of w, on the other hand, increases the ability of the learned classifier to generalize by decreasing

the propensity to over-fit to the training data. Parameter C is therefore a regularization parameter: larger values of C encourage small ξ^k and larger w, such that more points are classified correctly, but with a smaller margin.

4.1.1. The Feature Map and the Kernel Trick. Given a feature map ψ, the *feature vector* $\psi(x, y)$ for $x \in X$ and $y \in Y$ provides an alternate representation of the input-output pair (x, y). It is useful to consider feature maps ψ for which $\psi(x, y) = \phi(\chi(x, y))$, where $\chi : X \times Y \to \mathbb{R}^s$ for some $s \in \mathbb{N}$ is an *attribute map* that combines x and y into a single *attribute vector* $\chi(x, y)$ compactly representing the pair, and $\phi : \mathbb{R}^s \to \mathbb{R}^m$ for $m > s$ maps the attribute vector to a higher-dimensional space in a non-linear way. In this way, SVMs can achieve non-linear classification in the original space.

While we work hard to keep s small, the so-called *kernel trick* means that we do not have the same problem with m: it turns out that in the dual of Training Problem 1, $\psi(x, y)$ only appears in an inner product of the form $\langle \psi(x, y), \psi(x', y') \rangle$, or, for a decomposable feature map, $\langle \phi(z), \phi(z') \rangle$ where $z = \chi(x, y)$ and $z' = \chi(x', y')$. For computational tractability it therefore suffices that this inner product can be computed efficiently, and the "trick" is to choose ϕ such that $\langle \phi(z), \phi(z') \rangle = K(z, z')$ for a simple closed-form function K, known as the *kernel*.

In this paper, we consider *polynomial kernels* K_{polyd}, parameterized by $d \in \mathbb{N}^+$, and *radial basis function (RBF) kernels* K_{RBF}, parameterized by $\gamma = 1/(2\sigma^2)$ for $\sigma \in \mathbb{R}^+$:

$$K_{polyd}(z, z') = (z \cdot z')^d, \quad K_{RBF}(z, z') = \exp\left(-\gamma \left(\|z\|^2 + \|z'\|^2 - 2z \cdot z'\right)\right).$$

Both polynomial and RBF kernels use the standard inner product of their arguments, so their efficient computation requires that $\chi(x, y) \cdot \chi(x, y')$ can be computed efficiently.

4.1.2. Dealing with an Exponentially Large Output Space. Training Problem 1 has $\Omega(|Y|\ell)$ constraints, where Y is the output space and ℓ the number of training instances, and enumerating all of them is computationally prohibitive when Y is large. Joachims et al. [2009] address this issue for structural SVMs through constraint generation: starting from an empty set of constraints, this technique iteratively adds a constraint that is maximally violated by the current solution until that violation is below a desired threshold ϵ. Joachims et al. show that this will happen after no more than $O(\frac{C}{\epsilon})$ iterations, each of which requires $O(\ell)$ time and memory. However, this approach assumes the existence of an efficient separation oracle, which given a weight vector w and an input x finds an output $y \in \arg\max_{y \in Y} f_w(x, y)$. The existence of such an oracle remains an open question in application to combinatorial auctions; see Section 5.1.3 for additional discussion.

4.1.3. Required Information. In summary, the use of structural SVMs requires specification of the following:

(1) The input space X, the discrete output space Y, and examples of input-output pairs.
(2) An attribute map $\chi : X \times Y \to \mathbb{R}^s$. This function generates an attribute vector that combines the input and output data into a single object.
(3) A kernel function $K(z, z')$, typically chosen from a well-known set of candidates, e.g., polynomial or RBF. The kernel implicitly calculates the inner product $\langle \phi(z), \phi(z') \rangle$, e.g., between a mapping of the inputs into a high dimensional space.
(4) If the space Y is prohibitively large, a routine that allows for efficient separation, i.e., a function that computes $\arg\max_{y \in Y} f_w(x, y)$ for a given w, x.

In addition, the user needs to stipulate particular training parameters, such as the regularization parameter C, and the kernel parameter γ if the RBF kernel is being used.

4.2. Structural SVMs for Mechanism Design

We now specialize structural SVMs such that their learned discriminant function will manifest as a payment rule for a given symmetric outcome function g and distribution D. In this application, the input domain X is the space of type profiles Θ, and the output domain Y is the space Ω_1 of outcomes for agent 1. Thus we construct training data by sampling $\theta \sim D$ and applying g to these inputs: $\{(\theta^1, g_1(\theta^1)), \ldots, (\theta^\ell, g_1(\theta^\ell))\} = \{(\theta^1, o_1^1), \ldots, (\theta^\ell, o_1^\ell)\}$. For admissibility of the learned hypothesis $h_w(\theta) = \arg\max_{o_1 \in \Omega_1} w^T \psi(\theta, o_1)$, we require that $\psi(\theta, o_1) = (v_1(\theta_1, o_1), \psi'(\theta_{-1}, o_1))$. When learning payment rules, we therefore use an attribute map $\chi' : \Theta_{-1} \times \Omega_1 \to \mathbb{R}^s$ rather than $\chi : \Theta \times \Omega_1 \to \mathbb{R}^s$, and the kernel ϕ' we specify will only be applied to the output of χ'. This results in the following more specialized training problem:

$$\min_{w, \xi \geq 0} \frac{1}{2} w^T w + \frac{C}{\ell} \sum_{k=1}^{\ell} \xi^k \qquad \text{(Training Problem 2)}$$

$$\text{s.t. } (w_1 v_1(\theta_1^k, o_1^k) + w_{-1}^T \psi'(\theta_{-1}^k, o_1^k)) - (w_1 v_1(\theta_1^k, o_1) + w_{-1}^T \psi'(\theta_{-1}^k, o_1)) \geq \mathcal{L}(o_1^k, o_1) - \xi^k \quad \text{for all } k = 1, \ldots, \ell,\ o_1 \in \Omega_1$$

$$\xi^k \geq 0 \quad \text{for all } k = 1, \ldots, \ell.$$

If $w_1 > 0$ then the weights w together with the feature map ψ' define a price function $t_w(\theta_{-1}, o_1) = -(1/w_1) w_{-1}^T \psi'(\theta_{-1}, o_1)$ that can be used to define payments $p_w(\theta)$, as described in Section 3.1. In this case, we can also relate the regret in the induced mechanism (g, p_w) to the classification error as described in Section 3.3. Specifically, we can show that on any type profile θ^k of the training data, $rgt_1(\theta^k) \leq \xi^k / w_1$; see the full version of the paper for details.

We choose not to enforce $w_1 > 0$ explicitly in Training Problem 2, because adding this constraint leads to a dual problem that references ψ' outside of an inner product and thus makes computation of all but linear or low-dimensional polynomial kernels prohibitively expensive. Instead, in our experiments we simply discard hypotheses where the result of training is $w_1 \leq 0$. This is sensible since the discriminant function value should increase as an agent's value increases, and negative values of w_1 typically mean that the training parameter C or the kernel parameter γ is poorly chosen. It turns out that w_1 is indeed positive most of the time, and for every experiment a majority of the choices of C and γ yield positive w_1 values. For this reason, we do not expect the requirement that $w_1 > 0$ to be a problem in practice.[7]

4.2.1. Payment Normalization. One issue with the framework as stated is that the payments p_w computed from the solution to Training Problem 2 could be negative.

We solved this problem by normalizing payments, using a *baseline outcome* o_b: if there exists an outcome o' such that $v_1(\theta_1, o') = 0$ for every θ_1, this "null outcome" is used as the baseline; otherwise, we use the outcome with the lowest payment. Let $t_w(\theta_{-1}, o_1)$ be the price function corresponding to the solution w to Training Problem 2. Adopting the baseline outcome, the *normalized payments* $t'_w(\theta_{-1}, o_1)$ are defined as

$$t'_w(\theta_{-1}, o_1) = \max(0, t_w(\theta_{-1}, o_1) - t_w(\theta_{-1}, o_b)).$$

Note that o_b is only a function of θ_{-1}, even when there is no null outcome, so t'_w is still only a function of θ_{-1} and o_1.

4.2.2. Individual Rationality Violation. Even after normalization, the learned payment rule p_w may not satisfy IR. We offer three solutions to this problem, which can be used in combination.

[7] For multi-minded combinatorial auctions, $1049/1080 > 97\%$ of the trials had positive w_1, for the assignment problem all of the trials did; see Section 5 for details.

Payment offsets. One way to decrease the rate of IR violation is to add a payment offset, which decreases all payments (for all type reports) by a given amount. We apply this payment offset to all bundles other than o_b; as with payment normalization, the adjusted payment is set to 0 if it is negative.[8] Note that payment offsets decrease IR violation, but may increase regret.

Adjusting the loss function $\mathcal{L}$. We incur an IR violation when there is a null outcome o_{null} such that $g_1(\theta) \neq o_{null}$ and $f_w(\theta, o_{null}) > f_w(\theta, g_1(\theta))$ for some type θ, assuming truthful reports. This happens because $f_w(\theta, o_1)$ is a scaled version of the agent's utility for outcome o_1 under payments p_w. If the utility for the null outcome is greater than the utility for $g_1(\theta)$, then the payment $t_w(\theta_{-1}, g_1(\theta))$ must be greater than $v_1(\theta_1, g_1(\theta))$, causing an IR violation. We can discourage these types of errors by modifying the constraints of Training Problem 2: when $o_1^k \neq o_{null}$ and $o_1 = o_{null}$, we can increase $\mathcal{L}(o_1^k, o_1)$ to heavily penalize misclassifications of this type. With a larger $\mathcal{L}(o_1^k, o_1)$, a larger ξ^k will be required if $f_w(\theta, o_1^k) < f_w(\theta, o_{null})$. As with payment offsets, this technique will decrease IR violations but is not guaranteed to eliminate all of them. In our experimental results, we refer to this as the *null loss fix*, and the null loss refers to the value we choose for $\mathcal{L}(o_1^k, o_{null})$ where $o_1^k \neq o_{null}$.

Deallocation. In settings that have a null outcome and are *downward closed* (i.e., settings where a feasible outcome o remains feasible if o_i is replaced with the null outcome), we modify the function g to allocate the null outcome whenever the price function t_w creates an IR violation. This reduces ex post regret and in particular ensures ex post IR. On the other hand, the total value to the agents necessarily decreases under the modified allocation. In our experimental results, we refer to this as the *deallocation fix*.

5. APPLYING THE FRAMEWORK

In this section, we discuss the application of our framework to two domains: multi-minded combinatorial auctions and egalitarian welfare in the assignment problem.

5.1. Multi-Minded Combinatorial Auctions

A combinatorial auction allocates items $\{1, \ldots, r\}$ among n agents, such that each agent receives a possibly empty subset of the items. The outcome space Ω_i for agent i thus is the set of all subsets of the r items, and the type of agent i can be represented by a vector $\theta_i \in \Theta_i = \mathbb{R}^{2^r}$ that specifies its value for each possible bundle. The set of possible type profiles is then $\Theta = \mathbb{R}^{2^r n}$, and the value $v_i(\theta_i, o_i)$ of agent i for bundle o_i is equal to the entry in θ_i corresponding to o_i. We require that valuations are monotone, such that $v_i(\theta_i, o_i) \geq v_i(\theta_i, o_i')$ for all $o_i, o_i' \in \Omega_i$ with $o_i' \subseteq o_i$, and normalized such that $v_i(\theta_i, \emptyset) = 0$. Assuming agent symmetry and adopting the view of agent 1, the partial outcome rule $g_1 : \Theta \to \Omega_1$ specifies the bundle $g_1(\theta)$ allocated to agent 1; we require feasibility, so that no item is allocated more than once.

In a multi-minded CA, each agent is interested in at most b bundles for some constant b. The special case where $b = 1$ is called a single-minded CA. In our framework, the restriction to multi-minded CAs leads to a number of computational advantages. First, valuation profiles and thus the training data can be represented in a compact way, by explicitly writing down the valuations for the constant number of bundles each agent is interested in. Second, inner products between valuation profiles, which are required to apply the kernel trick, can be computed in constant time.

5.1.1. Attribute Maps. To apply structural SVMs to multi-minded CAs, we need to specify an appropriate attribute map χ. In our experiments we use two attribute maps

[8] It is again crucial that o_b depends only on θ_{-1}, so that the payment remains independent of θ_1 given o_1.

$\chi_1 : \Theta_{-1} \times \Omega_1 \to \mathbb{R}^{2^r(2^r(n-1))}$ and $\chi_2 : \Theta_{-1} \times \Omega_1 \to \mathbb{R}^{2^r(n-1)}$, which are defined as follows:

$$\chi_1(\theta_{-1}, o_1) = \begin{bmatrix} 0 \\ \cdots \\ 0 \\ \theta_{-1} \\ 0 \\ \cdots \\ 0 \end{bmatrix} \begin{matrix} \left.\vphantom{\begin{matrix} 0 \\ \cdots \\ 0 \end{matrix}}\right\} dec(o_1)(2^r(n-1)) \\ \\ \left.\vphantom{\begin{matrix} 0 \\ \cdots \\ 0 \end{matrix}}\right\} (2^r - dec(o_1) - 1)(2^r(n-1)), \end{matrix} \qquad \chi_2(\theta_{-1}, o_1) = \begin{bmatrix} \theta_2 \setminus o_1 \\ \theta_3 \setminus o_1 \\ \cdots \\ \theta_n \setminus o_1 \end{bmatrix}.$$

Here, $dec(o_1) = \sum_{j=1}^{r} 2^{j-1} \mathbf{I}_{j \in o_1}$ is a decimal index of bundle o_1, where $\mathbf{I}_{j \in o_1} = 1$ if $j \in o_1$ and $\mathbf{I}_{j \in o_1} = 0$ otherwise. Attribute map χ_1 thus stacks the vector θ_{-1}, which represents the valuations of all agents except agent 1, with zero vectors of the same dimension, where the position of θ_{-1} is determined by the index of bundle o_1. The resulting attribute vector is simple but potentially restrictive. It precludes two instances with different allocated bundles from sharing attributes, which provides an obstacle to generalization of the discriminant function across bundles. Attribute map χ_2 stacks vectors $\theta_i \setminus o_1$, which are obtained from θ_i by setting the entries for all bundles that intersect with o_1 to 0. This captures the fact that agent i cannot be allocated any of the bundles that intersect with o_1 if o_1 is allocated to agent 1.[9]

5.1.2. Efficient Computation of Inner Products. Inner products can be computed efficiently for both χ_1 and χ_2. A detailed discussion can be found in the full version of the paper.

5.1.3. Dealing with an Exponentially Large Output Space. Recall that Training Problems 1 and 2 have constraints for every training example (θ^k, o_1^k) and every possible bundle of items $o_1 \in \Omega_1$, of which there are exponentially many in the number of items in the case of CAs. In lieu of an efficient separation oracle, a workaround exists when the discriminant function has additional structure, such that the induced payment weakly increases as items are added to a bundle. Given this *item monotonicity*, it would suffice to include constraints for bundles that have a strictly larger value to the agent than any of their respective subsets. We further discuss this issue in the full version of the paper.[10]

5.2. The Assignment Problem

In the assignment problem, we are given a set of n agents and a set $\{1, \ldots, n\}$ of items, and wish to assign each item to exactly one agent. The outcome space of agent i is thus $\Omega_i = \{1, \ldots, n\}$, and its type can be represented by a vector $\theta_i \in \Theta_i = \mathbb{R}^n$. The set of possible type profiles is then $\Theta = \mathbb{R}^{n^2}$. We consider an outcome rule that maximizes *egalitarian welfare* in a lexicographic manner: first, the minimum value of any agent is maximized; if more than one outcome achieves the minimum, the second lowest value is maximized, and so forth. This outcome rule can be computed by solving a sequence of integer programs. As before, we assume agent symmetry and adopt the view of agent 1.

To complete our specification of the structural SVM framework for this problem, we need to define an attribute map $\chi_3 : \mathbb{R}^{n^2-n} \times \mathbb{N} \to \mathbb{R}^s$, where the first argument is the type profile of all agents but agent 1, the second argument is the item assigned to

[9] Both χ_1 and χ_2 are defined for a particular number of items and agents, and in our experiments we train a different classifier for each number of agents and items. In practice, one can pad out items and agents by setting bids to zero and train a single classifier.

[10] The experimental results in Section 6 do not assume item monotonicity and instead use a separation oracle that iterates over all possible bundles $o_1 \in \Omega_1$.

agent 1, and s is a dimension of our choosing. A natural choice for χ_3 is to set

$$\chi_3(\theta_{-1}, j) = (\theta_2[-j], \theta_3[-j], \ldots, \theta_n[-j]) \in \mathbb{R}^{(n-1)^2},$$

where $\theta_i[-j]$ denotes the vector obtained from θ_i by removing the jth entry. The attribute map thus reflects the agents' values for all items except item j, capturing the fact that the item assigned to agent 1 cannot be assigned to any other agent. Since the outcome space is very small, we choose not to use a non-linear kernel on top of this attribute vector.

6. EXPERIMENTAL EVALUATION

We perform a series of experiments to test our theoretical framework. To run our experiments, we use the *SVM^{struct}* package [Joachims et al. 2009], which allows for the use of custom kernel functions, attribute maps, and separation oracles.

6.1. Setup

We begin by briefly discussing our experimental methodology, performance metrics, and optimizations used to speed up the experiments.

6.1.1. Methodology. For each of the settings we consider, we generate three data sets: a training set, a validation set, and a test set. The training set is used as input to Training Problem 2, which in turn yields classifiers h_w and corresponding payment rules p_w. For each choice of the parameter C of Training Problem 2, and the parameter γ if the RBF kernel is used, a classifier h_w is learned based on the training set and evaluated based on the validation set. The classifier with the highest accuracy on the validation set is then chosen and evaluated on the test set. During training, we take the perspective of agent 1, so a training set size of ℓ means that we train an SVM on ℓ examples. Once a partial outcome rule has been learned, however, it can be used to infer payments for all agents. We exploit this fact during testing, and report performance metrics across all agents for a given instance in the test set.

6.1.2. Metrics. We employ three metrics to measure the performance of the learned classifiers. These metrics are computed over the test set $\{(\theta^k, o^k)\}_{k=1}^{\ell}$. *Classification accuracy* measures the accuracy of the trained classifier in predicting the outcome. Each instance of the ℓ instances has n agents, so in total we measure accuracy over $n\ell$ instances:[11]

$$\textit{accuracy} = 100 \cdot \frac{\sum_{k=1}^{\ell} \sum_{i=1}^{n} I(h_w(\theta_i, \theta_{-i}) = o_i^k))}{n\ell}.$$

Ex post regret sums over the ex post regret experienced by all agents in each of the ℓ instances in the validation set:

$$\textit{regret} = \frac{\sum_{k=1}^{\ell} \sum_{i=1}^{n} rgt_i(\theta_i^k, \theta_{-i}^k)}{n\ell}.$$

Individual rationality violation measures the fraction of individual rationality violation across all agents:

$$\textit{ir-violation} = \frac{\sum_{k=1}^{\ell} \sum_{i=1}^{n} I(irv_i(\theta_i, \theta_{-i}) > 0)}{n\ell}.$$

[11] For a given instance θ, there are actually many ways to choose (θ_i, θ_{-i}) depending on the ordering of all agents but agent i. We discuss a technique we refer to as sorting in Section 6.1.3, which will choose a particular ordering. When this technique is not used, for example in our experiments for the assignment problem, we simply fix an ordering of the other agents for each agent i and use the same ordering across all instances.

Table I. Results for multi-minded CA with optimal outcome rule, training set size 500

ζ	accuracy		regret		ir-violation	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
0.5	15.8	67.0	0.133	0.032	0.26	0.19
1.0	73.4	77.4	0.018	0.011	0.19	0.12
1.5	91.7	93.9	0.004	0.002	0.06	0.03

Table II. Results for multi-minded CA with greedy outcome rule, training set size 500

ζ	accuracy			regret			ir-violation		
	p_{vcg}	χ_1	χ_2	p_{vcg}	χ_1	χ_2	p_{vcg}	χ_1	χ_2
0.5	64.9	61.3	63.0	0.048	0.027	0.042	0.13	0.19	0.24
1.0	82.7	85.8	84.9	0.007	0.009	0.009	0.04	0.10	0.10
1.5	97.5	90.5	91.4	0.001	0.004	0.002	0.01	0.06	0.04

6.1.3. Optimizations. In the case of multi-minded CAs we map the inputs θ_{-1} into a smaller space, which allows us to learn more effectively with smaller amounts of data.[12] We use *instance-based normalization*, which normalizes the values in θ_{-1} by the highest observed value and then rescales the computed payment appropriately, and *sorting*, which orders agents based on bid values. These techniques are explained in more detail in the full version of the paper.

6.2. Single-Item Auction

As a sanity check, we perform experiments on the single-item auction with the optimal outcome rule, where the agent with the highest bid receives the item. We obtain excellent accuracy and very close approximation to the second-price payment rule. The framework is able to automatically learn the payment rule of Vickrey's auction. The complete results are deferred to the full version of the paper.

6.3. Multi-Minded CAs

We give a high-level overview of the type distribution and the two outcome rules used in the experiments, details can again be found in the full version of the paper.

The type distribution is inspired by Sandholm's decay distribution for single-minded CAs [Sandholm 2002], and is parameterized by two variables β and ζ: β controls the level of correlation between values of different agents, ζ controls the degree of complementarity between items.

The first outcome rule is the optimal rule g_{opt}, which selects a feasible allocation with maximum total value. It is well known that this outcome rule can be turned into a strategyproof mechanism (g_{opt}, p_{vcg}) by means of the Vickrey-Clarke-Groves payment rule p_{vcg}. The second outcome rule we experiment with is a generalization of the greedy allocation rule for single-minded CAs [Lehmann et al. 2002], which attempts to find an allocation with good welfare by greedily allocating bundles to agents based on a heuristic score. This rule can be made strategyproof in the special case of single-minded CAs, but not in the general multi-minded case.

We experiment with training sets of sizes 100, 300, and 500, and validation and test sets of size 1000. All experiments we report on are for a setting with 5 agents, 5 items, and 3 bundles per agent, and use $\beta = 0.5$, the RBF kernel, and parameters $C \in \{10^4, 10^5\}$ and $\gamma \in \{0.01, 0.1, 1\}$. Additional experimental results can be found in the full version of the paper.

6.3.1. Basic Results. Tables I and II present the basic results for multi-minded CAs with optimal and greedy outcome rules, respectively. For the greedy outcome rule we also present results for p_{vcg} as a baseline.[13] These results are not shown for the optimal outcome rule, where p_{vcg} has accuracy 100, regret 0, and IR violation 0.

[12] The barrier to using more data is not the availability of the data itself, but the time required for training, because training time scales quadratically in the size of the training set due to the use of non-linear kernels.

[13] The payment of an agent under the VCG-based payment rule p_{vcg} is equal to the marginal externality imposed by the agent on the other agents, relative to the outcome rule in question.

Table III. Impact of payment offset and null loss fix for $\zeta = 0.5$ and greedy outcome rule, training set size 300. All results are for χ_2, null loss values appear in the second row.

payment offset	accuracy			regret			ir-violation			ir-fix-welfare-avg		
	0.5	1.0	1.5	0.5	1.0	1.5	0.5	1.0	1.5	0.5	1.0	1.5
0	59.7	61.8	61.7	0.065	0.048	0.042	0.35	0.26	0.21	0.27	0.43	0.52
0.05	61.7	61.2	60.1	0.054	0.045	0.044	0.29	0.20	0.15	0.37	0.54	0.65
0.15	60.4	55.1	52.2	0.047	0.055	0.064	0.17	0.10	0.06	0.59	0.75	0.84
0.25	54.3	47.7	44.3	0.061	0.082	0.096	0.08	0.03	0.02	0.79	0.89	0.93

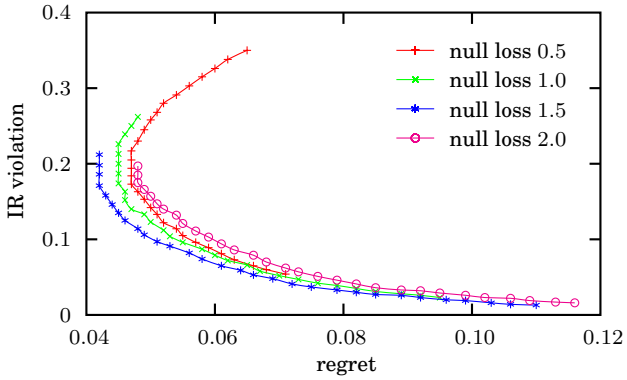

Fig. 1. Impact of payment offset and null loss fix for $\zeta = 0.5$ and greedy outcome rule, training set size 300

As expected, accuracy across all instances is negatively correlated with regret and ex post IR violation. Both the outcome rule and degree of complementarity between items controlled by ζ have a major effect on the results. Regret is higher for $\zeta = 0.5$, i.e., when complementarity between items is low, and χ_2 performs better for the optimal outcome rule while χ_1 performs better for the greedy outcome rule. For high complementarity, the greedy outcome rule tends to allocate all items to a single agent, and the learned payment rule sets high prices for small bundles to capture this property. For low complementarity, the allocation tends to be split and less predictable. Still, the best classifiers achieve average ex post regret of less than 0.032, for values normalized to $[0, 1]$, even though the corresponding prediction accuracy can be as low as 63%. For the greedy outcome rule, the performance of the learned payment rule is comparable to that of p_{vcg} when $\zeta \in \{1.0, 1.5\}$, and superior in the case $\zeta = 0.5$, where the difference between and greedy outcome rule and the optimal one is bigger.

6.3.2. Effect of Training Set Size. As expected, increasing the training set size from 100 to 300 to 500 leads to better results with higher accuracy and lower regret. Detailed results can be found in the full version of the paper.

6.3.3. IR Fixes. Table III summarizes our results regarding the various fixes to IR violations, for the particularly challenging case of the greedy outcome rule and $\zeta = 0.5$. The extent of IR violation decreases with larger payment offset and null loss. Regret tends to move in the opposite direction, but there are cases where IR violation and regret both decrease. The three rightmost columns of Table III list the average ratio between welfare after and before the deallocation fix, across the instances in the test set. With a payment offset of 0, a large welfare hit is incurred if we deallocate agents with IR violations. However, this penalty decreases with increasing payment offsets and increasing null loss. At the most extreme payment offset and null loss adjustment, the IR violation is as low as 2%, and the deallocation fix incurs a welfare loss of only 7%.

Table IV. Results for assignment problem with egalitarian outcome rule

n	accuracy				regret				ir-violation			
	vcg	tot-vcg	eg-vcg	p_w	vcg	tot-vcg	eg-vcg	p_w	vcg	tot-vcg	eg-vcg	p_w
2	64.3	67.5	67.5	89.0	0.018	0.015	0.015	0.023	0.03	0.01	0.01	0.03
4	40.6	43.1	30.8	71.0	0.111	0.123	0.199	0.054	0.07	0.09	0.03	0.02
6	27.1	29.9	20.0	59.0	0.189	0.208	0.290	0.074	0.10	0.13	0.03	0.01

Figure 1 shows a graphical representation of the impact of payment offsets and null losses. Each line in the plot corresponds to a payment rule learned with a different null loss, and each point on a line corresponds to a different payment offset. The payment offset is zero for the top-most point on each line, and equal to 0.29 for the lowest point on each line. Increasing the payment offset always decreases the rate of IR violation, but may decrease or increase regret. Increasing null loss lowers the top-most point on a given line, but arbitrarily increasing null loss can be harmful. Indeed, a null loss of 1.5 results in a slightly higher top-most point but significantly lower regret at this top-most point compared to a null loss of 2.0.

6.4. The Assignment Problem

In the assignment problem, agents' values for the items are sampled uniformly and independently from $[0, 1]$. We use a training set of size 600, validation and test sets of size 1000, and the RBF kernel with parameters $C \in \{10, 1000, 100000\}$ and $\gamma \in \{0.1, 0.5, 1.0\}$.

The performance of the learned payment rules is compared to that of three VCG-based payment rules. Let W be the total welfare of all agents other than i under the outcome chosen by g, and W_{eg} be the minimum value any agent other than i receives under this outcome. We then consider the following payment rules: (1) the *vcg* payment rule, where agent i pays the difference between the maximum total welfare of the other agents under any allocation and W; (2) the *tot-vcg* payment rule, where agent i pays the difference between the total welfare of the other agents under the allocation maximizing egalitarian welfare and W; and (3) the *eg-vcg* payment rule, where agent i pays the difference between the minimum value of any agent under the allocation maximizing egalitarian welfare and W_{eg}.

The results for attribute map χ_3 are shown in Table IV. We see that the learned payment rule p_w yields significantly lower regret than any of the VCG-based payment rules, and average ex post regret less than 0.074 for values normalized to $[0, 1]$. Since we are not maximizing the sum of values of the agents, it is not very surprising that VCG-based payment rules perform rather poorly. The learned payment rule p_w can adjust to the outcome rule, and also achieves a low fraction of ex post IR violation of at most 3%.

7. CONCLUSIONS

We have introduced a new paradigm for computational mechanism design in which statistical machine learning is adopted to design payment rules for given algorithmically specified outcome rules, and have shown encouraging experimental results. Future directions of interest include (1) an alternative formulation of the problem as a regression rather than classification problem, (2) constraints on properties of the learned payment rule, concerning for example the core or budgets, (3) methods that learn classifiers more likely to induce feasible outcome rules, so that these learned outcome rules can be used, (4) optimistically assuming item monotonicity and dropping constraints implied by it, thereby allowing for better scaling of training time with training set size at the expense of optimizing against a subset of the full constraints in the training problem, and (5) an investigation of the extent to which alternative goals such as regret percentiles or *interim* regret can be achieved through machine learning.

Acknowledgments

We thank Shivani Agarwal, Vince Conitzer, Amy Greenwald, Jason Hartline, and Tim Roughgarden for valuable discussions and the anonymous referees for helpful feedback. All errors remain our own. This material is based upon work supported in part by the National Science Foundation under grant CCF-1101570, the Deutsche Forschungsgemeinschaft under grant FI 1664/1-1, an EURYI award, and an NDSEG fellowship.

REFERENCES

ASHLAGI, I., BRAVERMAN, M., HASSIDIM, A., AND MONDERER, D. 2010. Monotonicity and implementability. *Econometrica 78,* 5, 1749–1772.

AUSUBEL, L. M. AND MILGROM, P. 2006. The lovely but lonely Vickrey auction. In *Combinatorial Auctions*, P. Cramton, Y. Shoham, and P. Steinberg, Eds. MIT Press, Chapter 1, 17–40.

BUDISH, E. 2010. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. Working Paper.

CAI, Y., DASKALAKIS, C., AND WEINBERG, S. M. 2012. An algorithmic characterization of multi-dimensional mechanisms. In *Proc. of 44th STOC*. Forthcoming.

CARROLL, G. 2011. A quantitative approach to incentives: Application to voting rules. Tech. rep., MIT.

CONITZER, V. AND SANDHOLM, T. 2002. Complexity of mechanism design. In *Proc. of 18th UAI Conference*. 103–110.

DAY, R. AND MILGROM, P. 2008. Core-selecting package auctions. *International Journal of Game Theory 36,* 3–4, 393–407.

ERDIL, A. AND KLEMPERER, P. 2010. A new payment rule for core-selecting package auctions. *Journal of the European Economic Association 8,* 2–3, 537–547.

GUO, M. AND CONITZER, V. 2010. Computationally feasible automated mechanism design: General approach and case studies. In *Proc. of 24th AAAI Conference*.

HARTLINE, J. D., KLEINBERG, R., AND MALEKIAN, A. 2011. Bayesian incentive compatibility via matchings. In *Proc. of 22nd SODA*. 734–747.

HARTLINE, J. D. AND LUCIER, B. 2010. Bayesian algorithmic mechanism design. In *Proc. of 42nd STOC*. 301–310.

JOACHIMS, T., FINLEY, T., AND YU, C.-N. J. 2009. Cutting-plane training of structural SVMs. *Machine Learning 77,* 1, 27–59.

LAHAIE, S. 2009. A kernel method for market clearing. In *Proc. of 21st IJCAI*. 208–213.

LAHAIE, S. 2010. Stability and incentive compatibility in a kernel-based combinatorial auction. In *Proc. of 24th AAAI Conference*. 811–816.

LAVI, R. AND SWAMY, C. 2005. Truthful and near-optimal mechanism design via linear programming. In *Proc. of 46th FOCS Symposium*. 595–604.

LEHMANN, D., O'CALLAGHAN, L. I., AND SHOHAM, Y. 2002. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM 49*, 577–602.

LUBIN, B. 2010. Combinatorial markets in theory and practice: Mitigating incentives and facilitating elicitation. Ph.D. thesis, Department of Computer Science, Harvard University.

LUBIN, B. AND PARKES, D. C. 2009. Quantifying the strategyproofness of mechanisms via metrics on payoff distributions. In *Proc. of 25th UAI Conference*. 349–358.

PARKES, D. C., KALAGNANAM, J., AND ESO, M. 2001. Achieving budget-balance with Vickrey-based payment schemes in exchanges. In *Proc. of 17th IJCAI*. 1161–1168.

PATHAK, P. AND SÖNMEZ, T. 2010. Comparing mechanisms by their vulnerability to manipulation. Working Paper.

RASTEGARI, B., CONDON, A., AND LEYTON-BROWN, K. 2011. Revenue monotonicity in deterministic, dominant-strategy combinatorial auctions. *Artificial Intelligence 175*, 441–456.

SAKS, M. AND YU, L. 2005. Weak monotonicity suffices for truthfulness on convex domains. In *Proc. of 6th ACM-EC Conference*. 286–293.

SANDHOLM, T. 2002. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence 135,* 1-2, 1–54.

TSOCHANTARIDIS, I., JOACHIMS, T., HOFMANN, T., AND ALTUN, Y. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research 6*, 1453–1484.